An evolution in the ShRep System

David Rodrigues
IST - Instituto Superior Técnico
L2F-Spoken Language Systems Laboratory - INESC-ID Lisboa
Rua Alves Redol 9, 1000-029 Lisboa, Portugal

David.Rodrigues@l2f.inesc-id.pt

Abstract. The starting point of this work was the ShRep system developed by João Graça. In it, João Graça presents a solution to solve some of the problems in the pipes-and-filters architecture, used mostly in natural language processing. The system replaces the pipes-and-filters architecture with a client-server architecture. In this architecture, the server acts as a repository in order to maintain all the data produced by the tools and to do all the processing necessary to the correct management of the data, while the client is composed by the natural language processing tools that request data from the repository, process it, and save the results in the repository.
After the MARv tool was fully integrated, the ShRep system presented a very high processing time. Because of this, some changes were made in order to optimize the system, namely by reducing the impact of the communication protocol between the client and the server. Finally a code model was developed in order to permit the use of the ShRep system in StandAlone mode; this was achieved by replacing the server with files, with the data being loaded at the beginning, and stored in the end of the tool processing cycle.

1 Introduction

Usually, Natural Language Processing (NLP) systems use pipeline architectures. This means that each tool processes data supplied by another tool and supplies the result to yet another tool of the chain. This process has some problems, related to the following:

- Each tool should know what do with the information that was received, and decide which of that information needs to be forwarded to the next tool;
- If some tools use different data types, conversion modules are required between then.

The first problem can be solved if each tool produces all the information required for the next tools, but this approach requires that each tool

should know the information needed by the other tools. Another approach consists in each tool forwarding all the information that was received to it. The later approach adds complexity to the system and performance penalties.

The second problem requires a conversion module between each tool that interchanges information with tools that have different data formats. In a chain with n elements, the system needs, at least, $n*n$ conversion modules.

The system ShRep - Shared Repository for Natural Language Processing [1], presents an alternative to the pipeline architecture. The architecture of the ShRep system is a framework using client-server architecture.

The server acts as a repository that maintains the data available for the tools and the relations between the layers of data. The data model defined in the server is present in figure 1 and is organized in layers, which allows differentiated analysis and signal data layers. The signal data layer allows us to describe data that could either be audio or text. The analysis layer is used to maintain results obtained by the tools that use the system; this layer also allows defined segmentations as well as defined classifications that add features to the segments. It's also possible to define relations between segments of the same analysis or between segments of different analysis.

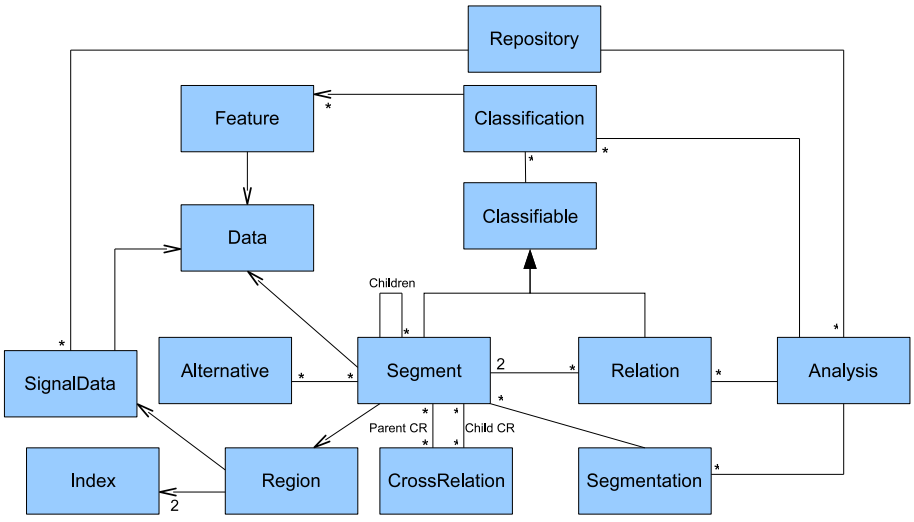

Fig. 1. Data model ShRep system.

The client side of the architecture presented in the ShRep system is the tool that interacts with the server in order to process the data. The system

provides some client libraries, assigned to some programming languages, which abstract the connection and the communication protocol between the tools and the server. This abstraction allows the developer of NLP tools to concern only with the use of the ShRep data model.

ShRep system solves the pipe and filters system first problem by keeping all the information readily available in the server side. One tool only pulls the information that it is needed to produce results. Nevertheless, even if one tool doesn't use all data existing in the repository, it is still available for use by other tools. The second presented problem for the pipe and filters system isn't eliminated in the ShRep system, but is certainly reduced. If a tool uses a different data model, conversion module is needed between the ShRep system and that tool, as well as another conversion module between the tool and the ShRep system. In an extreme case, if the system has n tools integrated, the system may need $2*n$ conversion modules.

The following tools were integrated in ShRep system, by João Graça:

- Sentence Splitter;
- Smorph which is a word tokenizer and a pos-tagger;
- PAsMo which is a rule-based rewriter.

In the development of this master thesis, two additional tools were integrated:

- MARv which is a morfo-sintactic disambiguator;
- Palavroso which is a word tokenizer and a pos-tagger, such as Smorph.

The MARv tool processed an XML file with 57 498 words and 102 776 classifications in 8.005 seconds. With the ShRep system as input, the MARv tool processed the same information in 22 minutes and 43.351 seconds (see table 1).

Parameter		XML File	ShRep System
Tempo	Real	8.005s	22m43s351
	User	7.688s	5m46s486
	SYS	0.314s	0m45s883

Table 1. Times for MARv tool, with ShRep system and XML file as input.

We detected that the implementation of the solution proposed by João Graça had some problems, mainly because the tool has very lengthy processing time when using the ShRep system, compared with the execution time of the same tool with XML files as input.

The analysis of the ShRep system shows that the biggest problem lies in the communication protocol, XML-RPC. The following problems were detected during the analysis:

- The information presented in the client is duplicated. For each request of a data model element, a request is made to the server without first verifying if the desired element already exists in the client. For instance, if a tool requests the same segment in n different moments, theclient requests the same segment n times;
- The client produces a server request in each request produced by one tool. For example, if a tool requires 50 000 segments with 150 000 classifications and creates 150 000 new classifications, the client performs 350 000 requests to the server;
- A large amount of information is transmitted in each request between the client to the server;

The proposed solution tries to reduce the problems detected on the communication protocol:

- Only one representation is kept on the client side, for each element that a tool requested from the server;
- Reduction in the number of requests made from the client to the server;
- Reduction in the amount of information transmitted on each request;

We also tried to eliminate the use of a communication protocol by converting the client into a standalone system.

1.1 Plan of the paper

The rest of this paper is organized as follows. Section 2 presents the proposed solution to reduce the problems in the communication protocol. Finally, Section 3 concludes and presents some lines for future work.

2 Solution

The difference of MARv tool in processing time between a XML file and ShRep system as input is associated with some problems existing in the ShRep system, mainly related with the communication protocol that bridges the client to the server.

2.1 Increase performance

The first step to solve these problems wat to implement a cache system in the client. The cache solves the data duplication problem, because the client first checks if an element is in the cache before executing a remote

call. The cache also reduces the number of remote calls made by the client, because if a tool needs the same element in two different moments, the client only makes one remote call and preserves that element in the cache.

In addition to the cache, all the information that was previously transmitted from the client to the server is replaced by identification information. For example, if a tool wants to add a classification to a given segment, the client only needs to send the identification of that segment and the directly related classification information. By using this identification instead transmitting all the unneeded information related with the segment, the amount of exchanged information is greatly reduced.

The next problem detected is that the client requests the server all the elements that are required by a tool. If an analysis requires 20 000 segments, the client makes 20 000 remote calls to the server side. To solve this problem, a new method was developed that in a single call, a set of elements is transfered from the server to the client, reducing the number of remote calls. These elements bring some related information aggregated, for instance, segments can bring its associated classifications. This solution requires that the client must have the capacity to keep all the segments sent by the server, and must give the correct segments to the tool.

We also tried to reduce the number of calls made by the client to the server. In the first version of ShRep system, every time a tool wanted to update some information in the server, it produces a remote call, that resulted in many calls to the server, each call with a high cost, in terms of processing time. The new solution accumulates a set of update requests in the client and only transmits them to the server when it has a significant amount of requests. The cost of transmitting more information in a single call is far less than the cost of making a larger number of calls. The implemented solution decreases the number of remote calls, but requires that the client overtakes some responsibilities that in the earlier version belonged to the server, such as:

- The client needs to attribute the new elements id's and to create relations between several elements in the domain model. This feature is important because the client may have some requests where the necessary information is only available on the client side. This occurs because the client only makes the update on the server when several update requests are made;
- The client needs to make all verifications that in the old version were only made by the server. For instance, one analysis can only be changed if its state is open;

With these functionalities implemented in the client, the main use of the server is to keep the data accessible to all other tools integrated in the system.

To aggregate the requests in the client each request is encapsulated with the Command Design Pattern [2]. The implementation of the com-

munication protocol has some limitations related to the amount of data transferred in one single call: the maximum number of commands that is possible to send from the client to the server is limited to 20 000.

We also reducted the amount of information transmitted between the client and the server, and between the server and the client. The information exchanged was very expressive, which results on lots of information to describe the data and lots of data itself. The objective is to reduce the amount of information that describes the data, using a compact version of the information. An example of this is presented in figure 2.

```
OLD VERSION
<create-segment-command id="1" type="" description="">
  <segmentation-identification id="1" analysis-id="1" />
  <region signal-data-id="1">
     <start-index><index type="0" value="1"></start-index>
     <end-index><index type="0" value="7"></end-index>
  </region>
</create-segment-command>

NEW VERSION
<create-segment-command s="1 0 0 1 1 1 1 0 1 0 7 0 0">
```

Fig. 2. Information reduction example.

Another change made in the ShRep system was the introduction of hash tables in the server, which reduces the time to search the domain elements that are referenced by an id.

With these five improvements the processing time of the MARv tool with the ShRep system was reduced by 98.03%, from the initial 22 minutes and 43.351 seconds to 26.862 seconds (see table 2). Although still larger than the use of the MARv tool with an xml file, it turns the use of the tool in the ShRep system possible.

Parameter		XML File	Start ShRep system	Final ShRep system
Tempo	Real	8.005s	22m43s351	0m26s862 (-98.03%)
	User	7.688s	5m46s486	0m9s265 (-97.33%)
	SYS	0.314s	0m45s883	0m0s440 (-99.04%)

Table 2. Times for MARv tool, with start and final ShRep system and XML file as input.

2.2 StandAlone version

Another experience made was to avoid the communication protocol. We developed a new version of the ShRep system, named standalone. In this version, the server is eliminated and the data is loaded and saved from a file. The idea of eliminating the communication protocol appeared because it was one of the biggest problem if the system. With the previous described changes, the client now has all the functionalities existing in the domain model of the server. So, the only change necessary in the client was the ability to work with files, in order to load the data for the initialization of the data model and to save the data when the tool finishes the processing. The only change in the tools is that, now, the name of the input files are given as input instead of the server location.

This version of ShRep system significantly reduces the process time, from 26.862s in the client/server version to 8.967s (see table 3). However, this version has a somewhat bad design that implies that all the data is loaded at the start of the process and is saved at the end of the process. An implementation where the system only loads the information that will be necessary for the tool and only saves the information that was created by the same tool would decrease the overall processing time, making it closer to the time obtained with the XML file.

Parameter		XML File	ShRep with Repository	ShRep *StandAlone*
Tempo	Real	7.508s	26.862s	8.967s
	User	7.253s	9.265s	8.777s
	SYS	0.277s	0.440s	0.123s

Table 3. Times for MARv tool, with ShRep system with repository and in standalone version and XML file as input.

.

3 Results and Future work

This works presents the ShRep system and validates it. The system was developed by João Graça, but an evaluation was still needed to allow the identification of existing problems. This work performs that evaluation, diagnoses some problems in the ShRep system and presents solutions that were implemented to solve them.

In conclusion, the changes made in the ShRep system turn it usable. The start version has a large processing time and the new version presents an acceptable time. This work also offer a new way to use the ShRep system, in a standalone version. This version presents a processing time

very close to the time presented by the tools using other kind of input data.

The integration of more tools in the system, the improvement of data loading/saving in the standalone version and the additional modifications to the communication protocol, such as the replacement of the XmlRpc by another communication protocol, are examples of possible improvements.

References

1. João de Almeida Varelas Graça. A framework for integrating natural language tools. Master's thesis, Universidade Técnica de Lisboa, Instituto Superior Técnico, 2006.
2. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns.* Addison-Wesley Professional, January 1995.